#16

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to:

Assistant Commissioner for Patents
Washington, D.C. 20231

On June 13, 2002

TOWNSEND and TOWNSEND and CREW LLP

By: Paula Tawle Hurley

PATENT
Attorney Docket No.: 020910-000310US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Examiner: Unassigned
Dan E. Rosenthal	Art Unit: 2123
Application No.: 10/053,348	PRELIMINARY AMENDMENT
Filed: November 2, 2001	
For: METHOD FOR ANALYTICAL JACOBIAN COMPUTATION IN MOLECULAR MODELING	

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

Prior to examination of the above-referenced application, please enter the following amendments and remarks.

IN THE SPECIFICATION:

Please replace the paragraph beginning on page 2, line 14, with the following rewritten paragraph:

The field of molecular modeling has successfully simulated the motion (molecular dynamics or MD) and determined energy minima or rest states (static

analysis) of many complex molecular systems by computers. Typical molecular modeling applications have included enzyme-ligand docking, molecular diffusion, reaction pathways, phase transitions, and protein folding studies. Researchers in the biological sciences and the pharmaceutical, polymer, and chemical industries are beginning to use these techniques to understand the nature of chemical processes in complex molecules and to design new drugs and materials accordingly. Naturally, the acceptance of these tools is based on several factors, including the accuracy of the results in representing reality, the size and complexity of the molecular systems that can be modeled, and the speed by which the solutions are obtained. Accuracy of many computations has been compared to experiment and generally found to be adequate within specified bounds. However, the use of these tools in the prior art has required enormous computing power to model molecules or molecular systems of even modest size to obtain molecular time histories of sufficient length to be useful.

Please replace the paragraph beginning on page 3, line 6, with the following rewritten paragraph:

Substantial work has been completed in reducing the computational load for molecular models, such as the reduction of model complexity by constraining higher order modes with rigid body assumptions, simplifying the model with rigid or flexible substructuring, Order(*N*) dynamics, efficient implicit solvent models, and multipole methods for the force field models (see, for example, U.S. Patent No. 5,424,963 on the commercial MBO(N)D software package). Co-pending applications, U.S. Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," and U.S. Appln. No. 10/053,354, entitled "METHOD FOR RESIDUAL FORM IN MOLECULAR MODELING," both of which are filed of even date, claim priority from the previously cited provisional patent applications and which are assigned to the present assignee, and which are incorporated by reference herein, describe further improvements in molecular models and numerical methods.

Please replace the paragraph beginning on page 3, line 19, with the following rewritten paragraph:

The primary reason molecular simulations are so slow is that currently used or applied numerical methods require very small timesteps, typically between 1 and 10 femtoseconds (10^{-15} to 10^{-14} seconds). Each timestep taken requires the computation of a new *state* (position and motion for each atom) of the particular molecular model, and then computation of the new set of forces resulting from the new state. For example, molecular dynamics simulations of the complex behavior of large molecules, such as the folding of a protein, typically need to cover a time span from at least a microsecond up to several seconds or even minutes. With techniques currently in common use, this results in the requirement to take 10^9 to 10^{16} timesteps in the computer simulation. The per-step computations of the state, and especially the forces, grow very expensive as the problem size increases. Even with the fastest computers available today, months, years or even centuries of computer time are required to solve such problems even for systems of modest size.

Please replace the paragraph beginning on page 4, line 6, with the following rewritten paragraph:

This common-sense belief is incorrect, however. The computer science sub-discipline of numerical analysis has produced an extensive theory of numerical integration for problems in which high frequencies exist but are of little interest. These problems are termed "stiff" problems (see, for example, Hairer and Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd ed.*, Springer, 1996). In these cases, it is the *stability* of the integration method, not the required solution *accuracy*, which limits the timestep. Integration methods exist which have *unconditional* stability, meaning that in theory they can take arbitrarily large

timesteps. Such methods have a mathematical property called "L-stability." Only so-called "implicit" integration methods *can* be L-stable, but very few implicit integration methods actually *are* L-stable. Previously cited co-pending U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," covers the use of implicit and in particular L-stable integration methods.

Please replace the paragraph beginning on page 4, line 24, with the following rewritten paragraph:

The same problem of high frequency molecular vibration for MD simulations causes problems for the calculation of Jacobians. For example, the repulsive van der Waal's forces that are generated as the electron orbitals of two atoms begin to overlap must be accounted for in a molecule. This quantum mechanical effect is often approximated by the so-called Lennard-Jones potential (Rapaport, *op. cit.*), which models the repulsive force as being proportional to $1/r^{13}$, where r is the distance between the centers of the atoms. This is an extremely stiff interatomic force which is characteristic of molecular dynamics (MD) simulations and poses particular difficulty for any numerical integration scheme used to advance time during the simulation. As a result and as mentioned previously, most prior art MD integration schemes have timesteps limited by the high frequencies generated by these extremely stiff repulsive forces. And in particular, the stiffness of the atomic forces greatly magnify the difficulty of forming certain required Jacobian matrices. Such Jacobian matrices are a necessary ingredient of any stable implicit integration scheme, such as described in the immediately cited co-pending application.

Please replace the paragraph beginning on page 5, line 24, with the following rewritten paragraph:

All general-purpose simulation codes provide routines to compute Jacobians numerically as follows. For a given equation to integrate $\dot{y} = f(y,t)$, the desired Jacobian is $J = \partial f / \partial y$ and is numerically computed:

$$J \approx \frac{\Delta f}{\Delta y}$$

where

$$\Delta f = f|_{y=y_2} - f|_{y=y_1}$$
$$\Delta y = y_2 - y_1$$

Note that the perturbation Δy has to be selected to give a good result and may be difficult to choose, especially for stiff systems. In contrast, analytic Jacobians are computed by solving directly, or in this case algorithmically, for the equation of the desired derivatives.

Please replace the paragraph beginning on page 6, line 8 , with the following rewritten paragraph:

The present invention provides for a method of modeling the behavior of a molecule. The method has the steps of selecting a torsion angle, rigid multibody model for the molecule, the model having equations of motion; selecting an implicit integrator; and generating an analytic Jacobian for the implicit integrator to integrate the equations of motion so as to obtain calculations of the behavior of the molecule. The analytic Jacobian J is derived from an analytic Jacobian of the Residual Form of the equations of motion and is described as:

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix and M is the mass matrix and ρ_u is the dynamic residual of the equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$. The method can also be used for any physical system which can be modeled by a torsion angle, rigid multibody system.

Please replace the paragraph beginning on page 9, line 1 , with the following rewritten paragraph:

The description of the preferred embodiment is divided into several sections. The first set of sections describes the MD simulation architecture, the multibody system (MBS) definitions, and Residual Form of the MBS equations for the subsequent descriptions. The next set of sections discusses the definition of the Jacobian, its role in the implicit integration method, and the computation of the analytic Jacobian using the Residual Form. Also shown is the superior accuracy and performance of the analytic Jacobian vs. the numerical Jacobian. Further efficiencies in the computation of the analytic Jacobian are discussed, specifically, exploiting the rigid body MBS to "contract" the size of the Jacobian matrix, and exploiting the topological structure of the MBS to eliminate unnecessary computations.

Please replace the paragraph beginning on page 9, line 29 , with the following rewritten paragraph:

In accordance with the present invention, a method for the solution of the equations of a molecular system is expressed in Residual Form to bypass the customary

step of producing the state derivatives directly. The Residual Form method has the following steps:

1) Discretization of the solution variables. The specific form of discretization is dictated by the particular implicit integration method used to advance the molecular model in time. Implicit integration follows from the Residual Form. Implicit integration, especially L-stable integrators and other highly stable integrators, such as implicit Euler, Radau5, SDIRK3, SDIRK4, other implicit Runge-Kutta methods, and DASSL or other implicit multistep methods, also provide other advantages for molecular modeling. See, for example, the above-cited U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," filed of even date. As a particularly simple example, when used with implicit Euler integration, the discretization is as follows:

$$\dot{q} = (q_n - q_{n-1})/h,\ \dot{u} = (u_n - u_{n-1})/h$$

where h is the timestep.

Please replace the paragraph beginning on page 11, line 15, with the following rewritten paragraph:

The general system architecture 48 of the software and some of its processes for modeling molecules in accordance with the present invention are illustrated in Fig. 1. Each large rectangular block represents a software module and arrows represents information which passes between the software modules. The software system architecture has a modeler module 50, a biochemistry components module 52, a physical model module 54, an analysis module 56 and a visualization module 58. The details of some of these modules are described below; other modules are available to the public.

Please replace the paragraph beginning on page 11, line 23, with the following rewritten paragraph:

The modeler module 50 provides an interface for the user to enter the physical parameters which define a particular molecular system. The interface may have a graphical or data file input (or both). The biochemistry components module 52 translates the modeler input for a particular mathematical model of the molecular system and is divided into translation submodules 60, 62 and 64 for mathematical modeling the molecule(s), the force fields and the solvent respectively of the system being modeled. There are several modeler and biochemistry components modules available including, for example, Tinker (Jay Ponder, TINKER User's Guide, Version 3.8, October 2000, Washington University, St. Louis, MO).

Please replace the paragraph beginning on page 11, line 32, with the following rewritten paragraph:

With the translated physical parameters from the biochemistry components module 52, the physical model module 54 defines the molecular system mathematically. At the core of the module 54 is a multibody system submodule 66. The analysis module 56, which communicates with the physical model module 54 and the visualization module 58, provides solutions to the computational models of the molecular systems defined by the physical model module 54. The analysis module 56 consists of a set of integrator submodules 68 which integrate the differential equations of the physical model module 54. The integrator submodules 68 advance the molecular system through time and also provide for static analyses used in determining the minimum energy configuration of the molecular system. The analysis module 56 and its integrator submodules 68 contain most of the subject matter of the present invention and are described in detail below.

Please replace the paragraph beginning on page 12, line 11, with the following rewritten paragraph:

The visualization module 58 receives input information from the biochemistry components module 52 and the analysis module 56 to provide the user with a three-dimensional graphical representation of the molecular system and the solutions obtained for the molecular system. Many visualization modules are presently available, an example being VMD (A. Dalke, *et al.*, VMD User's Guide, Version 1.5, June 2000, Theoretical Biophysics Group, University of Illinois, Urbana, Illinois).

Please replace the paragraph beginning on page 12, line 31 , with the following rewritten paragraph:

The MBS is an abstraction of the atoms and effectively rigid bonds that make up the molecular system being modeled and is selected to simplify the actual physical system, the molecule in its environment, without losing the features important to the problem being addressed by the simulation. With respect to the general system architecture illustrated in Fig. 1, the MBS does not include the electrostatic charge or other energetic interactions between atoms nor the model of the solvent in which the molecules are immersed. The force fields are modeled in the submodule 62 and the solvent in the submodule 64 in the biochemistry components module 52.

Please replace the paragraph beginning on page 13, line 31 , with the following rewritten paragraph:

The symbol for the vector from one point to another contains the name of the two points. Thus, r^{PQ} is the vector from the point P to point Q. A vector representing the velocity of a point in a reference frame contains the name of the point and the reference frame: ${}^{N}v^{P}$. Certain symbols to be introduced later relate two reference

frames. In this case, the symbol contains the name of two frames. Thus, ${}^{i}C^{k}$ is the direction cosine matrix for the orientation of frame k in frame i. This symbol refers to the direction cosine matrix for a typical body in its parent frame. Thus, ${}^{i}C^{k}(j)$ indicates the direction cosine matrix of the actual body j in question. The left and right superscripts do not change with the body index. This is also true for the other symbols. An asterisk indicates the transpose: $H^{*}(k)$, for example. A tilde over a vector indicates a 3 by 3 skew-symmetric cross product matrix: $\tilde{v}w \triangleq v \times w$. $\underline{\underline{E}}_{i}$ is an i by i identity matrix., and $\underline{0}_{i}$ is a zero vector of length i and $\underline{\underline{0}}_{i}$ is an i by i zero matrix.

Please replace the paragraph beginning on page 20, line 2 , with the following rewritten paragraph:

With the first kinematic calculations described above, the residual computation for the Residual Form method can be determined. A detailed description of the Residual Form and its application to molecular modeling is found in the previously cited co-pending U.S. Patent Appln. No. 10/053,354, entitled, "METHOD FOR RESIDUAL FORM IN MOLECULAR MODELING," filed of even date. This computation fills in two partitions of the vector $\begin{pmatrix} \rho_q \\ \rho_u \end{pmatrix}$ given previously. The first partition is called ρ_q, the kinematic residual, and the second partition is called ρ_u, the dynamic residual. The kinematic residual is computed from the difference between a $\dot{q}$, which is passed-in from the (implicit) integration submodules 66, and the derivative computed by each joint:

$$\dot{q} - W(q)u = \rho_q$$

Please replace the paragraph beginning on page 23, line 24 , with the following rewritten paragraph:

An important ingredient of this integration process is formation of the Jacobian of the differential equations. This is

$$J \triangleq \frac{\partial f}{\partial y}$$

Since the function f is itself computed by an algorithm rather than by an explicit formula, the Jacobian computation represents a substantial amount of work. In the simplest approach, the Jacobian can be formed numerically by differencing the derivative routine. This is a delicate operation because the quality of the Jacobian is a tradeoff between round-off and truncation errors. Typically half the working precision in the result is obtained by choosing a good perturbation size in the difference scheme. In practice, though, this is difficult to do.

Please replace the paragraph beginning on page 24, line 10 , with the following rewritten paragraph:

In general, G, the iteration matrix used in the Newton loop of the implicit integrator has the form $G = E - \alpha J$, where E is the identity matrix and α is some scalar function of the timestep. See the previously referenced U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," filed of even date, for a description of implicit integrators. Changes in step size require refactoring G, but not reforming J. Reforming J is needed only when the Jacobian is needed at a new state. G is used in a linear subproblem within a Newton loop. The following is solved:

$$G\Delta y = -r(y_n^i),$$
$$y_n^{i+1} = y_n^i + \Delta y$$

where $r(y)$ is the residual function for that particular implicit integration method.

Please replace the paragraph beginning on page 27, line 16 , with the following rewritten paragraph:

At a high level, the residual computation can be considered to depend upon two kinds of forces: 'motion forces' and external forces. The motion forces are computed directly by the multibody system. The external forces are available to the multibody system from a force modeling routine that computes the various interatomic forces such as electrostatics and solvents. A similar procedure is followed when computing the Jacobian. The multibody system builds the Jacobian of the motion forces, and combines it with the Jacobian of the external forces.

Please replace the paragraph beginning on page 27, line 31, with the following rewritten paragraph:

The same exchange occurs to compute Jacobians. The native Jacobians in their intrinsic coordinates are brought into the MBS coordinates. This requires the use of the chain-rule to transform between intrinsic and the MBS generalized coordinates. It is important that each effect co-computes its function value and Jacobian, because many of the same terms are needed for each computation. Each effect is transformed into a set of spatial loads $T_{effect}(k)$, where k is the index of a generic body in the system. The totality of these effects is given the symbol $T(k)$.

Please replace the paragraph beginning on page 34, line 3, with the following rewritten paragraph:

Note that the row-reduction procedure only needs to be done once before computing the Residual Jacobian. The overhead of performing the reduction is more than offset by the reduced cost of the smaller matrix vector products which must be formed.

Note that in forming $\frac{\partial T(k)}{\partial r(p,s)}$, there is no need to save the elements of the atomic Force Jacobian. That is, each element $\frac{\partial T(k,i)}{\partial r(p,s)}$ only needs to be available while its contribution to $\frac{\partial T(k)}{\partial r(p,s)}$ is being computed. So, more than one element of the big Force Jacobian is not required at a time.

Please replace the paragraph beginning on page 37, line 17, with the following rewritten paragraph:

1. Given (q,u) compute $z \triangleq -M^{-1}\rho_u(q,u,0)$ using the Direct Form method. Then set $\dot{u} = z$ and recompute $A(k)$, then ρ_u, which recomputes $\hat{T}(k)$.

Please replace the paragraph beginning on page 38, line 17, with the following rewritten paragraph:

Each interbody direction cosine matrix (and all the joint-specific) quantities depend only on the generalized coordinates of an individual joint. Thus, ${}^{i}dC^{k}(k)$ is nonzero only when the derivative is taken with respect to any of the coordinates for body k. To properly 'seed' the recursions being studied, a vector dq is passed in to the routine. For Jacobian computation we set one entry is set to 1, and all the other entries to 0. Then, the needed preliminary quantities are generated by a typical loop:

$$ {}^{i}dC^{k}(k) = \frac{\partial {}^{i}C^{k}(k)}{\partial q(k)} dq(k) \quad k = 1,\ldots n $$

The partial derivatives of the direction cosine matrices are generated analytically and displayed in the section, "Interface Contraction" above. These partial derivatives do *not* depend upon the particular column of the Jacobian that is being

computed. Setting a particular entry of dq to 1, and all the rest to 0 has the effect of annihilating the correct subset of the seed quantities.

$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k}$, the partial derivative of the interbody spanning vector is given by

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \lambda(k), \quad slider$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \underline{\underline{0}}_{3x4}, \quad ball$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \underline{\underline{0}}_{3}, \quad pin$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad free$$

$\lambda(k)$ here refers to the body's sliding axis which connects it to its parent body.

$\frac{\partial H(k)}{\partial q_k}$, the partial derivative of the joint map is

$$\frac{\partial H(k)}{\partial q_k} = \underline{\underline{0}}^{*}_{6}, \quad pin, slider$$

$$\frac{\partial H(k)}{\partial q_k} = \begin{bmatrix} \underline{\underline{0}}_3 & \underline{\underline{0}}_3 \end{bmatrix}, \quad ball$$

$$\frac{\partial H(k)}{\partial q_k} = \begin{bmatrix} \underline{\underline{0}}_3 & \underline{\underline{0}}_3 \\ \underline{\underline{0}}_3 & \dfrac{\partial {}^{i}C^{k}(k)}{\partial q_k} \end{bmatrix} for \quad k = 1,2,3,4, \quad free$$

$$\frac{\partial H(k)}{\partial q_k} = \underline{\underline{0}}_6 \; for \quad k = 5,6,7, \quad free$$

Please replace the paragraph beginning on page 39, line 13, with the following rewritten paragraph:

Since arbitrary perturbations to a set of Euler parameters do not produce a pure rotation, column contraction cannot be used when computing the corresponding column of the Jacobian. The row-reduced Force Jacobian can still (and must) be used.

Please replace the paragraph beginning on page 42, line 13 , with the following rewritten paragraph:

10. Back-solve the $\frac{\partial \rho_u}{\partial q}$ result of previous step with the mass-matrix to obtain the desired $\frac{\partial \dot{u}}{\partial q}$:

$$\frac{\partial \dot{u}}{\partial q} = -M^{-1}\frac{\partial \rho}{\partial q}$$

The back-solve operation is accomplished in the Direct Form method routine by processing a residual vector into a $\dot{u}$ vector. The Second Kinematics Calculations only needs to be performed once for the whole Jacobian, since the back-solves are done at the nominal value of the state. In fact, the Second Kinematics routine must have been called in Step 1 while computing z, so the variables should still be cached.

Please replace the paragraph beginning on page 44, line 7, with the following rewritten paragraph:

14. Back-solve the $\frac{\partial \rho_u}{\partial u}$ result of previous step with the mass-matrix to obtain the desired $\frac{\partial \dot{u}}{\partial u}$:

$$\frac{\partial \dot{u}}{\partial u} = -M^{-1}\frac{\partial \rho}{\partial u}$$

The back-solve operation is accomplished in the Direct Method routine by processing a

residual vector into a $\dot{u}$ vector. The Second Kinematics Calculations only need to be performed once, since the back-solves are done at the nominal value of the state. In fact, the Second Kinematics routine must have been called in Step 1 while computing z, so the variables should still be cached.

Please replace the paragraph beginning on page 47, line 21, with the following rewritten paragraph:

Therefore, while the foregoing is a complete description of some of the embodiments of the invention, it should be evident that various modifications, alternatives and equivalents may be made and used. Accordingly, the above description should not be taken as limiting the scope of the invention which is defined by the metes and bounds of the appended claims.

IN THE CLAIMS:

Please add new claims 17-44 as follows:

17. A method of modeling the behavior of a molecule, comprising selecting a torsion angle, rigid multibody model for said molecule; and generating a partition of an analytic Jacobian J for said model, said partition selected from the group comprising J_{qq}, J_{qu}, J_{uq} and J_{uu}, and

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}$$

where q are generalized coordinates, $\dot{q}$ are derivatives of said generalized coordinates with respect to time, u are generalized speeds, and $\dot{u}$ are derivatives of said generalized speeds with respect to time.

18. The method of claim 17 wherein said model has equations of motion and

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where W is a joint map matrix defining relationships between $\dot{q}$ and u, M is the system mass matrix defining relationships between $\dot{u}$ and the generalized forces on said bodies, ρ_u is the dynamic residual of said equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$.

19. The method of claim 18 wherein said molecule comprises a number of atoms and wherein said selecting step further comprises

reducing said number of atoms to a smaller number of rigid bodies so as to reduce the number of computations for Jacobian partitions J_{uq} and J_{uu}.

20. The method of claim 19 wherein said number of computations is reduced approximately by the average number of atoms in each of said rigid bodies.

21. A method of modeling the behavior of a physical system, comprising

selecting a torsion angle, rigid multibody model for said physical; and

generating a partition of an analytic Jacobian J for said physical model, said partition selected from the group comprising J_{qq}, J_{qu}, J_{uq} and J_{uu}, and

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}$$

where q are generalized coordinates, $\dot{q}$ are derivatives of said generalized coordinates with respect to time, u are generalized speeds, and $\dot{u}$ are derivatives of said generalized speeds with respect to time.

22. The method of claim 21 wherein said physical model has equations of motion and

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where W is a joint map matrix defining relationships between $\dot{q}$ and u, M is the system mass matrix defining relationships between $\dot{u}$ and the generalized forces on said bodies, ρ_u is the dynamic residual of said equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$.

23. Computer code for simulating the behavior of a molecule, said code comprising

a first module for a torsion angle, rigid multibody model for said molecule; and

a second module for generating a partition of an analytic Jacobian J for said model, said partition selected from the group comprising J_{qq}, J_{qu}, J_{uq} and J_{uu}, and

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}$$

where q are generalized coordinates, $\dot{q}$ are derivatives of said generalized coordinates with respect to time, u are generalized speeds, and $\dot{u}$ are derivatives of said generalized speeds with respect to time.

24. The computer code of claim 23 wherein said model has equations of motion, and

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where W is a joint map matrix defining relationships between $\dot{q}$ and u, M is the system mass matrix defining relationships between $\dot{u}$ and the generalized forces on said bodies, ρ_u is the dynamic residual of said equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$.

25. The computer code of claim 24 wherein said molecule comprises a number of atoms and wherein said first module further comprises

code reducing said number of atoms to a smaller number of rigid bodies so as to reduce the number of computations for Jacobian partitions J_{uq} and J_{uu}.

26. The computer code of claim 25 wherein said number of computations is reduced approximately by the average number of atoms in each of said rigid bodies.

27. Computer code for simulating the behavior of a physical system, said code comprising

a first module for a torsion angle, rigid multibody model for said physical system; and

a second module for generating a partition of an analytic Jacobian J for said model, said partition selected from the group comprising J_{qq}, J_{qu}, J_{uq} and J_{uu}, and

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}$$

where q are generalized coordinates, $\dot{q}$ are derivatives of said generalized coordinates with respect to time, u are generalized speeds, and $\dot{u}$ are derivatives of said generalized speeds with respect to time.

28. The computer code of claim 27 wherein said model has equations of motion, and

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where W is a joint map matrix defining relationships between $\dot{q}$ and u, M is the system mass matrix defining relationships between $\dot{u}$ and the generalized forces on

said bodies, ρ_u is the dynamic residual of said equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$

29. The method of claim 1 further comprising calculating said analytical Jacobian in no more than order N^{P-1} computations, where N is proportional to the number of rigid bodies in said model, and wherein a calculation of a numerical Jacobian for said model requires order N^P computations.

30. The method of claim 5 further comprising calculating said analytical Jacobian in no more than order N^{P-1} computations, where N is proportional to the number of rigid bodies in said model, and wherein a calculation of a numerical Jacobian for said model requires order N^P computations.

31. The computer code of claim 9 wherein said second module further comprises code for calculating said analytical Jacobian in no more than order N^{P-1} computations, where N is proportional to the number of rigid bodies in said model; and wherein a calculation of a numerical Jacobian for said model requires order N^P computations.

32. The computer code of claim 13 wherein said second module further comprises code for calculating said analytical Jacobian in no more than order N^{P-1} computations, where N is proportional to the number of rigid bodies in said model; and wherein a calculation of a numerical Jacobian for said model requires order N^P computations.

33. The method of claim 1 wherein a calculation of a numerical Jacobian for said model has an accuracy of K digits and further comprising

calculating said analytical Jacobian for said model with an accuracy of at least 2K digits.

34. The method of claim 5 wherein a calculation of a numerical Jacobian for said model has an accuracy of K digits and further comprising

calculating said analytical Jacobian for said model with an accuracy of at least 2K digits.

35. The computer code of claim 9 wherein a calculation of a numerical Jacobian for said model has an accuracy of K digits and said second module further comprises

code for calculating said analytical Jacobian for said model with an accuracy of at least 2K digits.

36. The computer code of claim 13 wherein a calculation of a numerical Jacobian for said model has an accuracy of K digits and said second module further comprises

code for calculating said analytical Jacobian for said model with an accuracy of at least 2K digits.

37. The method of claim 1 wherein said analytic Jacobian is derived from an analytic Jacobian of the Direct Form of the equations of motion.

38. The method of claim 37 wherein said analytic Jacobian J comprises

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \left(M^{-1}(f)\right)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \left(M^{-1}(f)\right)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix defining relationships between $\dot{q}$ and u, M^{-1} is the operational inverse of the system mass matrix M which defines relationships between $\dot{u}$ and the generalized forces f on said bodies.

39. The method of claim 5 wherein said analytic Jacobian is derived from an analytic Jacobian of the Direct Form of the equations of motion.

40. The method of claim 39 wherein said analytic Jacobian J comprises

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \text{ and } J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \left(M^{-1}(f)\right)}{\partial q} \text{ and } J_{uu} = \frac{\partial \left(M^{-1}(f)\right)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix defining relationships between $\dot{q}$ and u, M^{-1} is the operational inverse of the system mass matrix M which defines relationships between $\dot{u}$ and the generalized forces f on said bodies.

41. The computer code of claim 9 wherein said analytic Jacobian is derived from an analytic Jacobian of the Direct Form of the equations of motion.

42. The computer code of claim 41 wherein said analytic Jacobian J comprises

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \text{ and } J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \left(M^{-1}(f)\right)}{\partial q} \text{ and } J_{uu} = \frac{\partial \left(M^{-1}(f)\right)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix defining relationships between $\dot{q}$ and u, M^{-1} is the operational inverse of the system mass matrix M which defines relationships between $\dot{u}$ and the generalized forces f on said bodies.

43. The computer code of claim 13 wherein said analytic Jacobian is derived from an analytic Jacobian of the Direct Form of the equations of motion.

44. The computer code of claim 43 wherein said analytic Jacobian J comprises

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \text{ and } J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \left(M^{-1}(f)\right)}{\partial q} \text{ and } J_{uu} = \frac{\partial \left(M^{-1}(f)\right)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix defining relationships between $\dot{q}$ and u, M^{-1} is the operational inverse of the system mass matrix M which defines relationships between $\dot{u}$ and the generalized forces f on said bodies.

IN THE DRAWINGS:

Four sheets of replacement drawings are being submitted.

REMARKS

Consideration of the patent application, as preliminarily amended, is respectfully requested.

By this amendment of the specification, the applicant has corrected various typographical errors, filled in the blank spaces in the text of the specification with now issued U.S. Application Nos., and described his invention with greater precision. For example, the change in the mathematical expression on page 5 reflects the better statement that the numerically computed $\frac{\Delta f}{\Delta y}$ is approximately equal to the analytical expression of the Jacobian J. Likewise, the changes to the equation on page 39 make that equation more explicit with respect to the index k.

With respect to the claims, the applicant has also added new claims 17-44 to more particularly point out and distinctly claim the subject matter which he regards as his invention entitled. Claims 1-44 are now pending.

If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at 650-326-2400.

Respectfully submitted,

Joe Liebeschuetz
Reg. No. 37,505

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
Tel: (650) 326-2400
Fax: (650) 326-2422
JOL:pfh
PA 3227032 v1

VERSION WITH MARKINGS TO SHOW CHANGES MADE

The paragraph beginning on page 2, line 14, has been amended as follows:

The field of molecular modeling has successfully simulated the motion (molecular dynamics or [(] MD [)]) and determined energy minima or rest states (static analysis) of many complex molecular systems by computers. Typical molecular modeling applications have included enzyme-ligand docking, molecular diffusion, reaction pathways, phase transitions, and protein folding studies. Researchers in the biological sciences and the pharmaceutical, polymer, and chemical industries are beginning to use these techniques to understand the nature of chemical processes in complex molecules and to design new drugs and materials accordingly. Naturally, the acceptance of these tools is based on several factors, including the accuracy of the results in representing reality, the size and complexity of the molecular systems that can be modeled, and the speed by which the solutions are obtained. Accuracy of many computations has been compared to experiment and generally found to be adequate within specified bounds. However, the use of these tools in the prior art has required enormous computing power to model molecules or molecular systems of even modest size to obtain molecular time histories of sufficient length to be useful.

The paragraph beginning on page 3, line 6, has been amended as follows:

Substantial work has been completed in reducing the computational load for molecular models, such as the reduction of model complexity by constraining higher order modes with rigid body assumptions, simplifying the model with rigid or flexible substructuring, Order(*N*) dynamics, efficient implicit solvent models, and multipole methods for the force field models (see, for example, U.S. Patent No. 5,424,963 on the commercial MBO(N)D software package). Co-pending applications, U.S. Appln. No.

10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," and U.S. Appln. No. 10/053,354, entitled "METHOD FOR RESIDUAL FORM IN MOLECULAR MODELING," both of which are filed of even date, claim priority from the previously cited provisional patent applications and which are assigned to the present assignee, and which are incorporated by reference herein, describe further improvements in molecular models and numerical methods.

The paragraph beginning on page 3, line 19, has been amended as follows:

The primary reason molecular simulations are so slow is that currently used or applied numerical methods require very small timesteps, typically between 1 and 10 femtoseconds (10^{-15} to 10^{-14} seconds). Each timestep taken requires the computation of a new *state* (position and motion for each atom) of the particular molecular model, and then computation of the new set of forces resulting from the new state. For example, molecular dynamics simulations of the complex behavior of large molecules, such as the folding of a protein, typically need to cover a time span from at least a microsecond up to several seconds or even minutes. With techniques currently in common use, this results in the requirement to take 10^9 to 10^{16} timesteps in the computer simulation. The per-step computations of the state, and especially the forces, grow very expensive as the problem size increases. Even with the fastest computers available today, months, years or even centuries of computer time are required to solve such problems even for systems of modest size.

The paragraph beginning on page 4, line 6, has been amended as follows:

This common-sense belief is incorrect, however. The computer science sub-discipline of numerical analysis has produced an extensive theory of numerical integration for problems in which high frequencies exist but are of little interest. These problems are termed "stiff" problems (see, for example, Hairer and Wanner, *Solving*

Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd ed., Springer, 1996). In these cases, it is the *stability* of the integration method, not the required solution *accuracy*, which limits the timestep. Integration methods exist which have *unconditional* stability, meaning that in theory they can take arbitrarily large timesteps. Such methods have a mathematical property called "L-stability." Only so-called "implicit" integration methods *can* be L-stable, but very few implicit integration methods actually *are* L-stable. Previously cited co-pending U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," covers the use of implicit and in particular L-stable integration methods.

The paragraph beginning on page 4, line 24, has been amended as follows:

[But the] The same problem of high frequency molecular vibration for MD simulations causes problems for the calculation of Jacobians. For example, the repulsive van der Waal's forces that are generated as the electron orbitals of two atoms begin to overlap must be accounted for in a molecule. This quantum mechanical effect is often approximated by the so-called Lennard-Jones potential (Rapaport, *op. cit.*), which models the repulsive force as being proportional to $1/r^{13}$, where r is the distance between the centers of the atoms. This is an extremely stiff interatomic force which is characteristic of molecular dynamics (MD) simulations and poses particular difficulty for any numerical integration scheme used to advance time during the simulation. As a result and as mentioned previously, most prior art MD integration schemes have timesteps limited by the high frequencies generated by these extremely stiff repulsive forces. And in particular, the stiffness of the atomic forces greatly magnify the difficulty of forming certain required Jacobian matrices. Such Jacobian matrices are a necessary ingredient of any stable implicit integration scheme, such as described in the immediately cited co-pending application.

The paragraph beginning on page 5, line 24, has been amended as follows:

All general-purpose simulation codes provide routines to compute Jacobians numerically as follows. For a given equation to integrate $\dot{y} = f(y,t)$, the desired Jacobian is $J = \partial f / \partial y$ and is numerically computed:

$$J[=] \approx \frac{\Delta f}{\Delta y}$$

where

$$\Delta f = f|_{y=y_2} - f|_{y=y_1}$$
$$\Delta y = y_2 - y_1$$

Note that the perturbation Δy has to be selected to give a good result and may be difficult to choose, especially for stiff systems. In contrast, analytic Jacobians are computed by solving directly, or in this case algorithmically, for the equation of the desired derivatives.

The paragraph beginning on page 6, line 8 , has been amended as follows:

The present invention provides for a method of modeling the behavior of a molecule. The method has the steps of selecting a torsion angle, rigid multibody model for [said] the molecule, the model having equations of motion; selecting an implicit integrator; and generating an analytic Jacobian for the implicit integrator to integrate the equations of motion so as to obtain calculations of the behavior of the molecule. The analytic Jacobian J is derived from an analytic Jacobian of the Residual Form of the equations of motion and is described as:

$$J = \begin{pmatrix} \frac{\partial \dot{q}}{\partial q} & \frac{\partial \dot{q}}{\partial u} \\ \frac{\partial \dot{u}}{\partial q} & \frac{\partial \dot{u}}{\partial u} \end{pmatrix} \triangleq \begin{pmatrix} J_{qq} & J_{qu} \\ J_{uq} & J_{uu} \end{pmatrix}; \text{ and}$$

$$J_{qq} = \frac{\partial \dot{q}}{\partial q} = \frac{\partial (Wu)}{\partial q} \quad \text{and} \quad J_{qu} = \frac{\partial \dot{q}}{\partial u} = W$$

$$J_{uq} = \frac{\partial \dot{u}}{\partial q} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial q} \quad \text{and} \quad J_{uu} = \frac{\partial \dot{u}}{\partial u} = -M^{-1} \frac{\partial \rho_u(q,u,z)}{\partial u}$$

where q are the generalized coordinates, u are the generalized speeds, W is a joint map matrix and M is the mass matrix and ρ_u is the dynamic residual of the equations of motion, and z is $-M^{-1}\rho_u(q,u,0)$. The method can also be used for any physical system which can be modeled by a torsion angle, rigid multibody system.

The paragraph beginning on page 9, line 1, has been amended as follows:

The description of the preferred embodiment is divided into several sections. The first set of sections describes the MD simulation architecture, the multibody system (MBS) definitions, and Residual Form of the MBS equations for the subsequent descriptions. The next set of sections discusses the definition of the Jacobian, its role in the implicit integration method, and the computation of the analytic Jacobian using the Residual Form. Also shown is the superior accuracy and performance of the analytic Jacobian vs. the numerical Jacobian. Further efficiencies in the computation of the analytic Jacobian are discussed, specifically, exploiting the rigid body MBS to "contract" the size of the Jacobian matrix, and exploiting the topological structure of the MBS to eliminate unnecessary computations.

The paragraph beginning on page 9, line 29, has been amended as follows:

In accordance with the present invention, a method for the solution of the equations of a molecular system is expressed in Residual Form to bypass the customary step of producing the state derivatives directly. The Residual Form method has the following steps:

2) Discretization of the solution variables. The specific form of discretization is dictated by the particular implicit integration method used to advance the molecular model in time. Implicit integration follows from the Residual Form. Implicit integration, especially L-stable integrators and other highly stable integrators, such as implicit Euler, Radau5, SDIRK3, SDIRK4, other implicit Runge-Kutta methods, and DASSL or other implicit multistep methods, also provide other advantages for molecular modeling. See, for example, the above-cited U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," filed of even date. As a particularly simple example, when used with implicit Euler integration, the discretization is as follows:

$$\dot{q} = (q_n - q_{n-1})/h,\ \dot{u} = (u_n - u_{n-1})/h$$

where h is the timestep.

The paragraph beginning on page 11, line 15, has been amended as follows:

The general system architecture 48 of the software and some of its processes for modeling molecules in accordance with the present invention are illustrated in Fig. 1. Each large rectangular block represents a software module and arrows represents information which passes between the software modules. The software system architecture has a modeler module 50, a biochemistry components module 52, a physical model module 54, an analysis module 56 and a visualization module 58. The details of some of these modules are described below; other modules are available to the public.

The paragraph beginning on page 11, line 23, has been amended as follows:

The modeler module 50 provides an interface for the user to enter the physical parameters which define a particular molecular system. The interface may have a graphical or data file input (or both). The biochemistry components module 52 translates the modeler input for a particular mathematical model of the molecular system and is divided into translation submodules 60, 62 and 64 for mathematical modeling the molecule(s), the force fields and the solvent respectively of the system being modeled. There are several modeler and biochemistry components modules available including, for example, Tinker (Jay Ponder, TINKER User's Guide, Version 3.8, October 2000, Washington University, St. Louis, MO).

The paragraph beginning on page 11, line 32, has been amended as follows:

With the translated physical parameters from the biochemistry components module 52, the physical model module 54 defines the molecular system mathematically. At the core of the module 54 is a multibody system submodule 66. The analysis module 56, which communicates with the physical model module 54 and the visualization module 58, provides solutions to the computational models of the molecular systems defined by the physical model module 54. The analysis module 56 consists of a set of integrator submodules 68 which integrate the differential equations of the physical model module 54. The integrator submodules 68 advance the molecular system through time and also provide for static analyses used in determining the minimum energy configuration of the molecular system. The analysis module 56 and its integrator submodules 68 contain most of the subject matter of the present invention and are described in detail below.

The paragraph beginning on page 12, line 11, has been amended as follows:

The visualization module 58 receives input information from the biochemistry components module 52 and the analysis module 56 to provide the user with a three-dimensional graphical representation of the molecular system and the solutions obtained for the molecular system. Many visualization modules are presently available, an example being VMD (A. Dalke, *et al.*, VMD User's Guide, Version 1.5, June 2000, Theoretical Biophysics Group, University of Illinois, Urbana, Illinois).

The paragraph beginning on page 12, line 31, has been amended as follows:

The MBS is an abstraction of the atoms and effectively rigid bonds that make up the molecular system being modeled and is selected to simplify the actual physical system, the molecule in its environment, without losing the features important to the problem being addressed by the simulation. With respect to the general system architecture illustrated in Fig. 1, the MBS does not include the electrostatic charge or other energetic interactions between atoms nor the model of the solvent in which the molecules are immersed. The force fields are modeled in the submodule 62 and the solvent in the submodule 64 in the biochemistry components module 52.

The paragraph beginning on page 13, line 31, has been amended as follows:

The symbol for the vector from one point to another contains the name of the two points. Thus, r^{PQ} is the vector from the point P to point Q. A vector representing the velocity of a point in a reference frame contains the name of the point and the reference frame: ${}^{N}v^{P}$. Certain symbols to be introduced later relate two reference frames. In this case, the symbol contains the name of two frames. Thus, ${}^{i}C^{k}$ is the direction cosine matrix for the orientation of frame k in frame i. This symbol refers to the

direction cosine matrix for a typical body in its parent frame. Thus, ${}^{i}C^{k}(j)$ indicates the direction cosine of the actual body j in question. The left and right superscripts do not change with the body index. This is also true for the other symbols. An asterisk indicates the transpose: $H^{*}(k)$, for example. A tilde over a vector indicates a 3 by 3 skew-symmetric cross product matrix: $\tilde{v}w \triangleq v \times w$. $\underline{\underline{E}}_i$ is an i by i identity matrix., and $\underline{0}_i$ is a zero vector of length i and $\underline{\underline{0}}_i$ is an i by i zero matrix.

The paragraph beginning on page 20, line 2, has been amended as follows:

With the first kinematic calculations described above, the residual computation for the Residual Form method can be determined. A detailed description of the Residual Form and its application to molecular modeling is found in the previously cited co-pending U.S. Patent Appln. No. 10/053,354, entitled, "METHOD FOR RESIDUAL FORM IN MOLECULAR MODELING," filed of even date. This computation fills in two partitions of the vector $\begin{pmatrix} \rho_q \\ \rho_u \end{pmatrix}$ given previously. The first partition is called ρ_q, the kinematic residual, and the second partition is called ρ_u, the dynamic residual. The kinematic residual is computed from the difference between a $\dot{q}$, which is passed-in from the (implicit) integration submodules 66, and the derivative computed by each joint:

$$\dot{q} - W(q)u = \rho_q$$

The paragraph beginning on page 23, line 24, has been amended as follows:

An important ingredient of this integration process is formation of the Jacobian of the differential equations. This is

$$J \triangleq \frac{\partial f}{\partial y}$$

Since the function f is itself computed by an algorithm rather than by an explicit formula, the Jacobian computation represents a substantial amount of work. In the simplest approach, the Jacobian can be formed numerically by differencing the derivative routine. This is a delicate operation because the quality of the Jacobian is a tradeoff between round-off and truncation errors. Typically half the working precision in the result is [retained] obtained by choosing a good perturbation size in the difference scheme. In practice, though, this is difficult to do.

The paragraph beginning on page 24, line 10, has been amended as follows:

In general, G, the iteration matrix used in the Newton loop of the implicit integrator has the form $G = E - \alpha J$, where E is the identity matrix and α is [be] some scalar function of the timestep. See the previously referenced U.S. Patent Appln. No. 10/053,253, entitled "METHOD FOR LARGE TIMESTEPS IN MOLECULAR MODELING," filed of even date, for a description of implicit integrators. Changes in step size require refactoring G, but not reforming J. Reforming J is needed only when the Jacobian is needed at a new state. G is used in a linear subproblem within a Newton loop. The following is solved:

$$G\Delta y = -r(y_n^i),$$
$$y_n^{i+1} = y_n^i + \Delta y$$

where $r(y)$ is the residual function for that particular implicit integration method.

The paragraph beginning on page 27, line 16, has been amended as follows::

At a high level, the residual computation can be considered to depend upon two kinds of forces: 'motion forces' and external forces. The motion forces are computed directly by the multibody system. The external forces are available to the multibody system from a force modeling routine that computes the various interatomic forces such as electrostatics and solvents. A similar procedure is followed when computing the Jacobian. The multibody system builds the Jacobian of the motion forces, and combines it with the Jacobian of the external forces.

The paragraph beginning on page 27, line 31, has been amended as follows:

The same exchange occurs to compute Jacobians. The native Jacobians in their intrinsic coordinates are [be] brought into the MBS coordinates. This requires the use of the chain-rule to transform between intrinsic and the MBS generalized coordinates. It is important that each effect co-computes its function value and Jacobian, because many of the same terms are needed for each computation. Each effect is transformed into a set of spatial loads $T_{effect}(k)$, where k is the index of a generic body in the system. The totality of these effects is given the symbol $T(k)$.

The paragraph beginning on page 34, line 3, has been amended as follows:

Note that the row-reduction procedure only needs to be done once before computing the Residual Jacobian. The overhead of performing the reduction is more than offset by the reduced cost of the smaller matrix vector products which must be formed. Note that in forming $\frac{\partial T(k)}{\partial r(p,s)}$, there is no need to save the elements of the atomic Force Jacobian. That is, each element $\frac{\partial T(k,i)}{\partial r(p,s)}$ only needs to be available while its contribution

to $\frac{\partial T(k)}{\partial r(p,s)}$ is being computed. So, more than one element of the big Force Jacobian is not required at a time.

The paragraph beginning on page 37, line 17, has been amended as follows:

2. Given (q,u) compute $z \triangleq -M^{-1}\rho_u(q,u,0)$ using the Direct Form method. [Also] Then v set $\dot{u} = z$ and recompute $A(k)$, then ρ_u, which recomputes $\hat{T}(k)$.

The paragraph beginning on page 38, line 17, has been amended as follows:

Each interbody direction cosine matrix (and all the joint-specific) quantities depend only on the generalized coordinates of an individual joint. Thus, ${}^{i}dC^{k}(k)$ is nonzero only when the derivative is taken with respect to any of the coordinates for body k. To properly 'seed' the recursions being [studying] studied, a vector dq is passed in to the routine. For Jacobian computation we set one entry is set to 1, and all the other entries to 0. Then, the needed preliminary quantities are generated by a typical loop:

$$ {}^{i}dC^{k}(k) = \frac{\partial {}^{i}C^{k}(k)}{\partial q(k)} dq(k) \quad k = 1,\ldots n $$

The partial derivatives of the direction cosine matrices are generated analytically and displayed in the section, "Interface Contraction" above. These partial derivatives do *not* depend upon the particular column of the Jacobian that is being computed. Setting a particular entry of dq to 1, and all the rest to 0 has the effect of annihilating the correct subset of the seed quantities.

$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k}$, the partial derivative of the interbody spanning vector is given by

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \lambda(k), \quad slider$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \underline{\underline{0}}_{3x4}, \quad ball$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \underline{0}_3, \quad pin$$

$$\frac{\partial r^{Q_i Q_k}(k)}{\partial q_k} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad free$$

$\lambda(k)$ here refers to the body's sliding axis which connects it to its parent body.

$\frac{\partial H(k)}{\partial q_k}$, the partial derivative of the joint map is

$$\frac{\partial H(k)}{\partial q_k} = \underline{0}^*_6, \quad pin, slider$$

$$\frac{\partial H(k)}{\partial q_k} = \begin{bmatrix} \underline{\underline{0}}_3 & \underline{\underline{0}}_3 \end{bmatrix}, \quad ball$$

$$\frac{\partial H(k)}{\partial q_k} = \begin{bmatrix} \underline{\underline{0}}_3 & \underline{\underline{0}}_3 \\ \underline{\underline{0}}_3 & \dfrac{\partial {}^i C^k(k)}{\partial q_k} \end{bmatrix} \text{for } k = 1,2,3,4 \quad free$$

$$\frac{\partial H(k)}{\partial q_k} = \underline{\underline{0}}_6 \text{ for } k = 5,6,7, \quad free$$

The paragraph beginning on page 39, line 13, has been amended as follows:

Since arbitrary perturbations to a set of Euler parameters do not produce a pure rotation, column contraction cannot be used when computing the

corresponding column of the Jacobian. The row-reduced Force Jacobian can still (and must) be used.[.]

The paragraph beginning on page 42, line 13, has been amended as follows:

10. Back-solve the $[\ \frac{\partial \rho}{\partial q}\ ]\ \underline{\frac{\partial \rho_u}{\partial q}}$ result of previous step with the mass-matrix to obtain the desired $\frac{\partial \dot{u}}{\partial q}$:

$$\frac{\partial \dot{u}}{\partial q} = -M^{-1}\frac{\partial \rho}{\partial q}$$

The back-solve operation is accomplished in the Direct Form method routine by processing a residual vector into a $\dot{u}$ vector. The Second Kinematics Calculations only needs to be performed once for the whole Jacobian, since the back-solves are done at the nominal value of the state. In fact, the Second Kinematics routine must have been called in Step 1 while computing *z*, so the variables should still be cached.

The paragraph beginning on page 44, line 7, has been amended as follows:

14. Back-solve the $[\frac{\partial \rho}{\partial u}\ ]\ \underline{\frac{\partial \rho_u}{\partial q}}$ result of previous step with the mass-matrix to obtain the desired $\frac{\partial \dot{u}}{\partial u}$:

$$\frac{\partial \dot{u}}{\partial u} = -M^{-1}\frac{\partial \rho}{\partial u}$$

The back-solve operation is accomplished in the Direct Method routine by processing a residual vector into a $\dot{u}$ vector. The Second Kinematics Calculations only need to be performed once, since the back-solves are done at the nominal value of the state. In fact,

the Second Kinematics routine must have been called in Step 1 while computing *z*, so the variables should still be cached.

The paragraph beginning on page 47, line 21, has been amended as follows:

Therefore, while the foregoing is a complete description of some of the embodiments of the invention, it should be evident that various modifications, alternatives and equivalents may be made and used. Accordingly, the above description should not be taken as limiting the scope of the invention which is defined by the metes and bounds of the appended claims.